Description

Automation system with automation objects with a directory structure and method for the management of automation objects in a directory structure

The invention relates to an automation system which has at least one automation object.

An automation system of this type is used in particular in the area of automation technology. An automation system of this type generally comprises a multiplicity of individual automation objects, which are frequently highly dependent on the automation object of the engineering system respectively used. This has the consequence that automation objects of one manufacturer often require their own engineering system and cannot be used in other systems with automation objects of other manufacturers.

Robert Orfali et al: "The Essential Distributed Objects Survival Guide", 1996, John Wiley & Sons Inc., New York, USA, XP002152444, discloses the standardized middleware CORBA, which allows location-, platform- and implementation-independent communication between applications. The CORBA Version 2.0 makes it possible for messages be exchanged between Object Request Brokers (ORB) of various manufacturers and in particular also over the Internet. An ORB makes it possible for a client to send a message transparently to a server object, the server object being able to run on the same machine or another machine. The ORB is responsible for finding the server object, calling up the function there, transferring the parameters and returning the result to the client.

The invention is based on the object of specifying an automation system which makes it possible for automation solutions to be created on a parallel and/or distributed basis.

This object is achieved by an automation system with the features specified in claim 1.

The invention is based on the realization that in previous solutions the data of the automation solution are generally stored in a central data store such as a database

system. The data storage system then controls the access of various users to the data. In this case, it is ensured that each user only sees consistent data and is isolated from changes made by other users. This generally takes place by a user being granted exclusive access to his required data. In this time, these data are not available to other users for working on them. Therefore, this solution has the following disadvantages:

- **No parallel working:** users can only work on the same data records one after the other.
- **Slow exchange of partial results:** results only become usable for other users when the data have been released again by the last person working on them.
- **No joint working:** a number of users cannot work on the same objects together and exchange interim results.

The solution according to the invention permits immediate and permanent access to currently created partial solutions by the special way in which the directory is structured as a directory service. The directory service provides all developers with access to the current partial solutions and automation objects. This results in the following advantages:

- **Parallel working:** users can work on the same data records, required for different tasks (for example interconnection and parameterization), on a parallel basis.
- **Immediate availability of partial results:** results become usable for other users more quickly, not only when the data are released again by the last person working on them.
- **Joint working:** a number of users can work on the same objects together and exchange interim results.
- **Distributed working:** users can work on a (spatially) distributed basis; by means of the directory, they can, if need be, always re-synchronize the stages

they have reached in working.

The invention is described in more detail and explained below on the basis of the exemplary embodiments represented in the figures, in which:

figure 1 shows a basic representation of how a directory is structured and its entries and

figure 2 shows a schematic representation of the use of the directory entries.

Figure 1 shows a basic representation of how a directory is structured and its entries. The automation system has a directory V, in which object names O1..On of automation objects can be stored. Each object name O1..On is assigned a directory entry, which contains first information data O11 for an object reference, second information data O12 as a list of the modules contained in the automation object, third information data (O13) for the identification of interface data and fourth information data (O14) with names of subcomponents.

With the aid of the directory structure shown in figure 1, references to created (partial) solutions and/or automation objects are stored with descriptive data. As in a telephone book, the name of the object can be used to find its reference (i.e. its telephone number).

Along with a reference to the actual object, the entry comprises a description of its technological functionality through the list of names of the modules contained, a listing of the names of any subcomponents and a description of its interface, which makes it possible for other objects/tools to use the objects referenced in this way.

Figure 2 a schematic representation of the use of the directory entries. After the creation of an object, it is entered at certain points in time in the directory as entry OE1 for a first automation object. It can then be viewed by other users/tools. They can then use the name to request a reference to the object and work on or copy the latter directly.

Entering or changing or removing an object entry in the directory does not have to take place instantaneously. Here, too, the analogy with a telephone book again applies: even if individual entries become invalid, as a whole it can still be used. This property is important in particular in the case of distributed working, since the communication expenditure is minimized in this way. If an object is still in the directory, but no longer available, this is indicated when it is attempted to request a copy.

To sum up, the invention consequently relates to an automation system which has at least one automation object 1, with a directory V for storing object names O1..On of the automation objects, an object name O1..On being assigned a directory entry Oe1..Oen which has first information data O11 as a reference to the automation object, second information data O12 as a description of the technological functionality and third information data O13 as a description of interfaces of the automation object. This results in immediate and permanent access to currently created (partial) solutions, so that parallel and/or distributed working on automation objects is possible.